Automatic Training Data Cleaning for Text Classification

Hassan H. Malik
Thomson Reuters
195 Broadway, New York, NY
hassan.malik@thomsonreuters.com

Vikas S. Bhardwaj
Thomson Reuters
195 Broadway, New York, NY
vikas.bhardwaj@thomsonreuters.com

***Abstract*—Supervised text classification algorithms rely on the availability of large quantities of quality training data to achieve their optimal performance. However, not all training data is created equal and the quality of class-labels assigned by human experts may vary greatly with their levels of experience, domain knowledge, and the time available to label each document. In our experiments, focused label validation and correction by expert journalists improved the Micro and Macro-F_1 scores achieved by Linear SVMs by as much as 14.5% and 30% respectively, on a corpus of professionally labeled news stories.**

Manual label correction is an expensive and time consuming process and the classification quality may not linearly improve with the amount of time spent, making it increasingly more expensive to achieve higher classification quality targets. We propose ATDC, a novel evidence-based training data cleaning method that uses training examples with high-quality class-labels to automatically validate and correct labels of noisy training data. A subset of these instances are then selected to augment the original training set. On a large noisy dataset with about two million news stories, our method improved the baseline Micro-F_1 and Macro-F_1 scores by 9% and 13% respectively, without requiring any further human intervention.

I. Motivation

Document classification is a task that involves assigning a document to one or more applicable categories from a set of pre-defined categories, based on its contents. In some domains, document classification serves as primary means to prevent information overload. For example, professional investors in the financial industry commonly subscribe to news categories of their interest, allowing them to receive a small fraction of relevant news stories from tens of thousands of new stories delivered over news feeds every day. In these domains, inaccurate document classification may have a high business impact because it prevents documents from reaching their target audiences.

A. Classification Inconsistencies in Manual Document Classification

Documents may be classified manually by human experts or automatically by machines. Manual document classification is often subject to a high degree of inconsistency. For example, a single news desk may include a combination of junior, mid-career and 30-year veteran journalists. Each journalist typically specializes in one or two subject areas and may have varying degrees of knowledge about categories outside those areas. Since a significant percentage of news stories belong to multiple subject areas, high-quality labeling is only possible if each story is reviewed by a team of journalists that has expertise in all subject areas that relates to the story. However, this ideal condition is rarely met in real-life because when a news-worthy event happens, getting the story out as quickly as possible always takes the highest priority. This often means that the journalist responsible for writing the story is also solely responsible for assigning appropriate categories to the story, as involving multiple experts may be time or resource prohibitive.

Automatic document classification systems typically use supervised classification algorithms with a computational complexity that is linear in the document length[1]. These algorithms are efficient and inherently more consistent than humans in assigning documents to categories[2]. However, since these algorithms learn classification models from a set of manually-classified training documents, human labeling inconsistency may also have a direct negative impact on their classification performance.

We have conducted an experiment that clearly demonstrates this problem. We obtained training and test sets by evenly splitting[3] about 12,700 news stories that were selected by journalists to train an automatic news classification system. Existing manually-assigned class-labels were also available for all stories in this dataset. A team of senior journalists that included experts from all relevant subject areas reviewed existing classification for each news story and corrected classification mistakes, i.e., by removing wrong class-labels and adding missing class-labels.

We first used the original manually-assigned class-labels to train binary Linear SVMs for each of the 198 classes that were used in this experiment. The regularization parameter C was automatically selected from the set $\{10^k | k = -4, ..., 2\}$ using a 5-fold cross-validation on the given training data. The trained classifiers were then applied to the

[1] Classification-time complexity for Linear SVMs, Naïve Bayes, and many other popular algorithms

[2] Unlike human experts, a pre-trained system is likely to assign a document to the same set of categories regardless of the document arrival time or the number of documents processed concurrently

[3] Using a pseudo-random selection scheme

Table I
CLASSIFICATION PERFORMANCE OF BINARY SVMS USING THE ORIGINAL AND REVISED CLASS-LABELS ON A DATASET OF NEWS STORIES.

	Original class-labels	Revised class-labels
Micro-F_1	0.592	0.678
Macro-F_1	0.511	0.662

test set, and Micro and Macro-F_1 scores were computed using the original class-labels as ground truth for the test set. Next, we retrained the classifiers using the revised class-labels for the same set of training documents, and also used the revised class-labels as ground truth for the test set. Table I summarizes the results of this experiment. We observe that higher-quality labels improved the Micro and Macro-F_1 scores by as much as 14.5% and 30% respectively. The improvement in Macro-F_1 score was more significant than the Micro-F_1 score because journalists may have a better understanding of frequently-used categories that are outside their own subject areas as compared to infrequent, highly specialized categories, resulting in more classification mistakes with respect to infrequently-used categories.

B. Augmenting High-Quality Labeled Data with Noisy-Labeled Data

While high-quality class-labels certainly seems to be useful in improving the classification performance of supervised classification algorithms, they may be quite expensive to obtain. In our experiments, experts needed an average of about 5 minutes per story to review and correct class-labels, or over a thousand man hours for the dataset used in Table I, when stories were presented to them along with their existing class-labels in a tool that was specifically developed for this purpose. In addition, the overall classification quality may not linearly improve with the amount of time spent, making it increasingly more expensive to achieve higher classification quality targets.

Since documents with low-quality "everyday" class-labels may be readily available in abundance at many organizations (and from publicly-available web-based databases), we are motivated to explore if these documents could be used in conjunction with some high-quality labeled data to further improve the classification quality, without requiring additional human intervention. However, the simple approach of directly augmenting high-quality labeled training data with low-quality labeled data may only hurt the classification performance.

To demonstrate this problem, we incrementally augmented the training dataset with revised high-quality class-labels from the prior experiment with randomly selected news stories from the same news archive in batches of about 25,000 stories, and retrained the classifiers. The new classifiers were then applied to the original test set, using the revised high-quality class-labels as ground-truth for evaluating the classification performance. Figure 1 presents the results of this experiment. Classification performance consistently decreased as additional news stories with low-quality class-labels were added to the training set, and the noise completely dominated the results when about 150,000 stories were added.

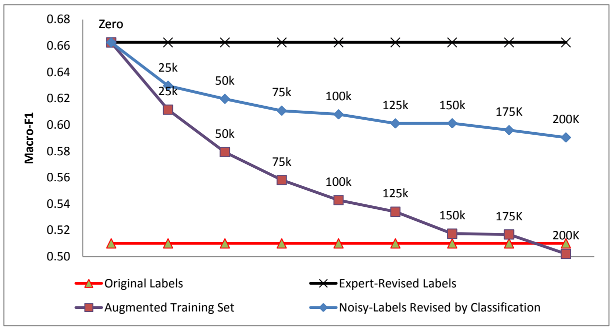

Figure 1. The change in Macro-F_1 performance as batches of news stories with low-quality class-labels were added to a training set with high-quality class-labels (line with squares), and when the noisy class-labels are revised by classification and then added to the same training set (line with diamonds). Horizontal lines on the top and bottom of the graph represents baseline Macro-F_1 scores with high-quality and low-quality class-labels, respectively (Table I). The test set remained constant in all experiments.

Instead of directly adding noisy-labeled documents to a training set with high-quality class-labels, it may seem intuitive to revise their classification using a set of classifiers trained on documents with high-quality class-labels, before adding these documents to the training set. To validate this intuition, we trained a set of classifiers using the same training set with high-quality class labels and then used these classifiers to revise the class-labels of noisy-labeled documents from the same news archive. As we show in Figure 1, this approach significantly limits the loss in classification performance but still does not improve over the baseline classification performance achieved using the high-quality training data at its own.

In this paper we propose a novel evidence-based training data cleaning method that uses documents with high-quality class-labels to automatically validate and correct labels of documents with noisy class-labels. A subset of the corrected documents is then selected to augment the original training set. Using the dataset with revised high-quality class-labels from Table I and about two million documents with noisy class-labels from the same corpus, our method improved the baseline Micro-F_1 and Macro-F_1 scores by 9% and 13% respectively. Unlike existing TDC methods [1], [4], [6], our method does not require any further human intervention.

II. RELATED WORK

Automatic text classification is a well-studied research problem. We cover a few representative methods here and refer the reader to [13] for a comprehensive survey.

Supervised text classifiers are typically constructed by learning a model using previously labeled documents and then applying this model to obtain labels for previously unseen documents [10]. Semi-supervised approaches have also been used for text classification and were found useful when the labeled dataset is small but a comparatively large set of unlabeled data is available [12], [3]. The news domain typically consists of multi-label documents. Multi-label classification is commonly achieved by training a separate binary classifier for each class and then applying all trained classifiers to an incoming document (or a subset of these classifiers in case of hierarchical classification [13]). A few methods that focus on multi-label classification include [11], [15].

Training Data Cleaning (TDC) has been studied extensively for many machine learning tasks. In computational linguistics, TDC methods primarily focused on identifying annotation errors in training data. Some work has also focused on analyzing annotator behavior to identify potentially incorrect annotations. Bhardwaj et al. [2] used probabilistic methods to identify annotators that are outliers among the group. Annotations performed by these annotators are either automatically discarded, or selected as candidates for manual revision, to improve the data quality for word-sense annotation. Sheng et al. [14] proposed a crowd-sourcing method that uses multiple untrained annotators to re-label the same data and then combines the results to correct annotation errors. In the realm of text classification, Esuli and Sebastiani [4] proposed a boosting-based method for multi-label data that combines several independent binary classifiers and finds misclassified documents as candidates for manual re-annotation. Fukumoto and Suzuki [6] proposed a method that first trains a SVM classifier and then removes all support vectors that the SVM has identified, from the training set. A Naïve Bayes classifier is then trained on the modified training set, and used to reclassify the removed support vectors. Any support vectors whose original label does not match the newly assigned label are considered mislabeled.

Existing TDC techniques are indeed useful in identifying labeling errors. However, they do not provide an automated way of correcting these errors and require re-labeling of the identified documents by human experts. Our work improves on the existing research by automatically identifying noisy-labeled documents that may augment good-quality training data to improve the classification performance. It automatically corrects labeling mistakes in these documents, and may also find missing labels in a completely automated fashion, eliminating the need for further human intervention. To the best of our knowledge, our work is the first attempt in this promising direction.

III. Automatic Training Data Cleaning

In the section we describe our automatic training data cleaning method. First, we formally define the problem and the terms used in the rest of the paper. We then discuss a key observation and finally discuss steps involved in our method.

A. Preliminaries

Let C be a set of class-labels, H be a set of documents such that each document H_i is associated with a set of high-quality class-labels HL_i, and $HL_i \in \{\emptyset, C\}$, Let L be a set of documents such that each document L_i is associated with a set of low-quality class-labels LL_i, and $LL_i \in \{\emptyset, C\}$.

Since we are primarily concerned with improving the classification performance over the baseline performance achieved with H alone, and $L \gg H$ for all practical purposes, there is no need to try finding accurate classification for each document in L. Instead, we focus on finding a subset S of L that may augment H to achieve this goal. For each document S_i, we aim to find a set of class-labels SL_i, and $SL_i \in \{C\}$.

B. *"Absence of evidence is not evidence of absence"*

In supervised text classification, all instances that are not labeled positive for a class are generally considered negative for it. Drawing inspiration from the famous quote by Dr. Karl Sagan[4], we argue that this assumption should only be made for a given document when there is explicit evidence available about both the positivity and the negativity of the document with respect to all available classes.

For example, the team of experts that had revised class-labels for the dataset used in Section I-A was explicitly tasked with classifying each document to all applicable classes, and documents were reviewed by multiple experts to ensure completeness. In such situations, the negativity assumption may be safely made because the chances of missing any positive label(s) for a given document are very low. In contrast, if the class-labels are known to be noisy, the positivity assumption is likely to be safer (or at-least more manageable) than the negativity assumption. Therefore, we use each document H_i in H as a positive example for all classes in HL_i, and as a negative example for the rest of the classes, but use each document S_i in S only as a positive example for classes in SL_i, and not as a negative example for any class in the remainder of our experiments. Section IV-E demonstrates the significant impact of this decision on classification performance.

C. Method

We make the following observations.

- Since L is noisy-labeled, each class-label c that exists in the set of class-labels LL_i for a document L_i may be considered as a weak evidence w that L_i belongs to c

[4] "Absence of evidence is not evidence of absence", Chapter 12, "The Demon-Haunted World"

- We may independently use H to generate additional weak evidence e about how L_i relates to c
- If e validates w, and if L_i improves the classification performance of c on a set of test documents from H, when used as a positive example for c to train a new classifier along with existing training documents from H and S, L_i may be considered as a candidate for inclusion in S as a positive example for c
- Some highly-specialized classes in C might be rarely used in L. For such classes, we may need to solely rely on evidence from H to find additional documents from L for inclusion in S

Figure 2 uses these observations to find additional positive examples for classes in C from L. It begins by splitting the available high-quality training data into two sets and reporting the baseline classification performance. Set TRN is used to find additional documents from L, and TST is used to evaluate the change in classification performance once S is completely populated.

Next, we split TRN into k sets and the available noisy-labeled documents L into n sets, where k and n are user-defined parameters. Splitting TRN into multiple sets allows us to generate independent evidence from a subset of TRN (i.e., set TD) to validate the noisy class-labels in L, select candidates for inclusion in S, and use the remainder of TRN (i.e., set SD) to estimate the incremental lift in classification performance without using documents in TST. Whereas splitting L into smaller sets allows us to process a large noisy-labeled dataset in small batches on a machine with limited computational resources.

We then iterate k times and in each iteration, use the corresponding TD and SD to evaluate the noisy-labeled documents on a class-by-class basis (Lines 9-33). For each class C_x, we first train a binary classifier B using the high-quality training data TD and obtain the baseline classification quality score on SD (lines 10-11). Any appropriate evaluation metric may be used for this purpose; we have used class F_1 scores in our experiments. B also serves as an evidence generation source, as we discuss in the next paragraph.

Next, we initialize a list Z (line 12) to hold the new positive candidates for C_x. Then for each batch L_j of noisy-labeled documents, we first obtain a set of clusters CLU by applying an unsupervised clustering algorithm on the documents in TD and L_j together (Line 14). We then obtain two sets of evidences from TD about how each document in L_j relates to C_x. The first set of evidences is obtained by applying B (i.e., a binary classifier trained on TD) to documents in L_j (Line 15), producing a positive or negative result for each document on L_j with respect to C_x. The second set of evidences is obtained by analyzing clusters in CLU where each document in L_j occurs, and counting the number of neighbors with high-quality class labels (i.e., from TD) that are marked positive for C_x (Line 16).

```
Require: A set $C$ of classes, a set $L$ of sparse noisy-labeled documents,
    a set $H$ of sparse documents with high-quality class-labels, and a
    set $P$ of evidence evaluation schemes pre-sorted in the decreasing
    order of scheme strictness; a parameter $k \geq 2$ and a parameter $n \geq 1, n \leq |L|$.
 1: Split $H$ into two sets, a training set $TRN$ and a test set $TST$.
 2: Use documents in $TRN$ to train binary classifiers for all classes in
    $C$, apply these classifiers to $TST$ and report the baseline classification performance
 3: Split $TRN$ into $k$ sets.
 4: Split $L$ into $n$ sets.
 5: $S = \{\emptyset\}$
 6: for $i = 1, \ldots, k$ do
 7:   $TD$ = all documents in sets $TRN_{0 \ldots i-1}$ and $TRN_{i+1 \ldots k}$
 8:   $SD$ = documents in $TRN_i$
 9:   for $x = 1, \ldots, |C|$ do
10:     $B$ = a binary classifier for $C_x$ trained using documents in $TD$
11:     $MQ = evaluate(apply(B, SD))$
12:     $Z = \{\emptyset\}$
13:     for $j = 1, \ldots, n$ do
14:       $CLU$ = clusters obtained by clustering documents in $TD$
          and $L_j$ together
15:       $ES1 = apply(B, L_j)$
16:       $ES2 = getHighQualityNeighbourCount(L_j, TD, CLU)$
17:       for $y = 1, \ldots, |P|$ do
18:         $M = select(L_j, ES1, ES2, P_y, Z)$
19:         $B1$ = a binary classifier for $C_x$ trained using documents
            in $TD$, $Z$ and $M$, using $TD$ as positive and negative,
            and $Z$ and $M$ as positive only
20:         $CQ = evaluate(apply(B1, SD))$
21:         if $CQ \geq MQ$ then
22:           Append $M$ to $Z$ storing $P_y$ as auxiliary information
23:           $MQ = CQ$
24:         end if
25:       end for
26:       for $u = 1, \ldots, |Z|$ do
27:         if Document $Z_u$ does not exist in $S$ then
28:           Append $Z_u$ to $S$
29:         end if
30:         Append $C_x$ to the set of class labels for document $Z_u$
            in $S$, while keeping track of the strictest evidence evaluation scheme that had selected $Z_u$ for $C_x$ as auxiliary
            information
31:       end for
32:     end for
33:   end for
34: end for
35: $S = selectSubset(S)$
36: Use $TRN$ and $S$ to train binary classifiers for all classes in $C$
    (Section III-B), apply these classifiers to $TST$ and report the classification performance
37: return $S$
```

Figure 2. Automatic training data cleaning

For each noisy-labeled document d in L_j, we now have three evidences indicating how it relates to C_x. Firstly, the original (noisy) class-labels assigned to d may or may not contain C_x. Then, the classification evidence may mark d as either positive or negative for C_x. Lastly, the clustering evidence produces a value of 0 or higher indicating the number of documents with high-quality class-labels that shared a cluster with d and contained C_x in their list of class-labels.

We apply a series of evidence evaluation schemes, in the decreasing order of their strictness, to select candidates from L_j for inclusion in Z (Lines 17-25). Method $select$ (Line 18) takes L_j, the available evidences, an evidence evaluation scheme, and the current members of Z as input and returns a subset of L_j that is not in Z, and satisfies the evidence evaluation scheme. A new binary classifier $B1$ for C_x is then trained using the selected candidates M, documents in TD, and Z, using documents in TD as positive and negative instances for C_x, and documents in Z and M as positive only for C_x (Section III-B). If the new classifier improves upon the current quality, the selected documents are added to Z while storing the evidence evaluation scheme and it's corresponding parameters as auxiliary information.

We have used the following evidence evaluation schemes. These schemes are used to accept a document d as positive for C_x. The Clustering (Clust.) evidence is considered positive if the evidence contains a value greater than or equal to t, where t is user-defined.

Scheme	$C_x \in$ Orig. Labels	Positive Evidence
S1	Yes	Classifier and Clust.
S2	Yes	Classifier or Clust.
S3	No	Classifier and Clust.

Once a class is processed, all documents in Z are added to S as positive examples for C_x (Lines 26-30). Since the same document may be selected as a positive example for the same class by different TD and SD combinations, we keep track of the strictest evidence evaluation scheme that had selected the document for the class.

Finally, new classifiers are trained on TRN and the selected subset of S (Line 35-36) and the classification performance on the hold-out test dataset TST is computed.

IV. Empirical Evaluation

A. Experimental Setup

Our noisy-labeled dataset L consisted of English news stories from January 2006 to December 2010 published by Reuters. Existing noisy, manually-assigned class-labels were available for all stories in this dataset and each story was originally classified against at-least one of the classes used in our experiments. The high-quality labeled dataset H consisted of about 12,700 stories. In order to prevent any overlap between H and L, we first filtered L to eliminate all stories in H using their unique story identifiers, and then used cosine similarity to identify and eliminate stories in L that were highly similar to any story in H (using 85% cosine similarity as threshold), leaving about 1.97 million unique news stories in $|L|$. We dropped highly similar stories because journalists often reuse large chunks of existing news stories as background knowledge, which may result in selecting stories that over-fit the validation set but fail on unseen data.

The set of class-labels C assigned to the documents were the leaf nodes of a classification hierarchy and $|C| = 198$ (we have avoided the internal nodes to simplify the experimental setup). H was split into 2 equal sets, TRN and TST for training and testing respectively. The same training and test sets were also used for experiments in Section I. The parameters $k = 5$ and $n = 100$ were used for splitting TRN and L into subsets. The selected value of n results in processing noisy-labeled documents in batches of about 20,000 documents. These values for k and n were selected intuitively and tuning these values may further improve the classification performance.

Our experiments used Linear SVMs to train binary classifiers, selected based on their success on a variety of text classification problems [13], [16], [8]. However, it is important to note that our method does not depend on SVMs and we expect it to complement any supervised classification algorithm. To train Linear SVMs in Linear-time, we have used the LibLinear library [5]. The overall classification performance was evaluated using two standard multi-label evaluation metrics. Micro-F_1 globally computes a single F_1 score regardless of the classes, and therefore favors large classes. In contrast, Macro-F_1 is computed as an average of per-class harmonic mean of precision and and recall scores, and favors small classes.

We have evaluated two different classes of clustering algorithms to obtain secondary evidence for the class-labels assigned to documents in L. We used two variants of k-Means (i.e., Spherical k-means and Kernel k-means using the TCT [7] implementation) to explore the conventional neighborhood-based clustering techniques. The number of clusters k was set to obtain clusters that averaged 100 documents (i.e., if clustering 1000 documents, $k = 10$). We additionally used a more recent clustering algorithm, IDHC [9], which is different than k-means in that it does not take a parameter k and produces a variable number of clusters.

We then applied the ATDC algorithm (Section III-C) to obtain a set of documents $S \in L$ with revised labels, and to learn a new classifier using TRN and S and analyze its performance on TST.

B. Analyzing the Classification Performance with Additional Training Data from S

We conducted two different sets of experiments to analyze the improvement in the quality of classification, by incrementally adding more data from S to TRN. We also compare these results across different clustering algorithms.

In the first experiment, we sorted the documents in S in the decreasing order of the evidence evaluation scheme strictness (Section III-C) for each class, and then in each iteration of the experiment, selected the top m documents for each class, varying m from 10 to 2000. A new set of classifiers was trained with the selected documents and TRN, and applied on TST. Figure 3 presents the change in Micro and Macro F_1 scores as we varied m, for each clustering algorithm.

We observe that as m increases, classification performance initially increases, and then drops slightly. This indicates that taking a subset of the selected documents (in this case, Top-100 documents) that was selected by more strict evidence evaluation schemes yields the best classification performance. Comparing these results with the baseline results in Table I, we observe that the additional documents improved the Micro F_1 score from 0.678 to about 0.74 and Macro-F_1 score from 0.66 to about 0.75, an improvement of about 9% and 13% respectively. We also observe that all clustering algorithms performs about the same, with IDHC having a slight edge over the others.

To verify the superior quality of choosing documents in S selected by stricter evidence validation schemes, we conducted another experiment, in which we repeated the process of incrementally adding more data from S to TRN (Figure 4) but unlike the previous experiment, the documents were selected in a random order and not in the order of most-to-least strict evidence validation scheme.

From Figure 4, we can thus conclude that documents selected by stricter schemes are more reliable for training. This also makes intuitive sense as the strictness of a scheme is directly proportional to the amount of evidence available to assign a label to the document.

C. Comparing Various Evidence Evaluation Schemes

We also conducted an experiment to compare the different evidence evaluation schemes and to analyze the performance improvement by considering each scheme individually. We divided the documents in S into $|P|$ sets according to the scheme that selected the class-label, and trained $|P|$ classifiers by combining each set with TRN (Figure 5). Since all clustering algorithms yielded similar performance, we only plotted the scores for IDHC in Figure 5.

From Figure 5, we observe that the stricter schemes that considers the evidence from both the classifier and the document clusters, and have higher thresholds for considering cluster-based evidence, performs the best. The stricter schemes S1-8, S1-4 and S3-8 alone surpass the baseline score in table I, and achieves an improvement of about 10% in classification performance. The other schemes performed close to the baseline F_1 scores.

D. Comparing Against an Existing Data Cleaning Method

Our method automatically identifies and corrects labeling errors, and also selects a subset of the corrected documents that are likely to improve the classification performance on unseen data, when combined with high-quality training data. In the absence of a directly-comparable method, we adapted the text TDC method proposed by Esuli and Sebastiani [4] to identify 'good' labels (non-erroneous) in noisy-labeled documents, and then compared the incremental classification performance achieved by documents with all 'good' labels with the documents selected by our method.

More specifically, we selected L_1, a subset of L that contains about 200,000 documents (the same subset used in Figure 1). Using the implementation available from the first authors' website [4], we used MP-Boost to learn a classifier from TRN and to identify a set L_2 with 'clean' documents from L_1 (i.e. documents whose original label(s) agreed with the classifier assigned label(s)). We then retrained the classifier using both TRN and L_2 as training data and applied the resulting classifier to TST. Similarly, we applied our method on the same datasets to select a subset of L_1 with automatically-cleaned labels (using IDHC to generated clustering-based evidence), added the selected documents to TRN, trained a classifier using TRN and the selected documents, and applied the resulting classifier on TST.

Figure 6 presents the results of this experiment. We observe that the Macro-F_1 scores achieved by the MP-Boost-based method dropped from the baseline score of 0.64 to 0.6 (about 6.25% decrease) as we increased the amount of presumably cleaned data added to TRN, whereas using our method, the performance increased from 0.68 to 0.73 (or about 7.3%). Note that the difference in baselines scores is due to the difference in classification algorithms used by the two methods, but the incremental gain or loss in classification performance are independent of the classification algorithm used, as SVM also yielded similar results when used to revised labels of the same noisy dataset (Figure 1).

E. Comparing the Positivity and Negativity Assumptions

To support the argument in section III-B, we selected the subset of documents from S that performed the best in the top-k experiment using IDHC (section IV-B), using all selected class-labels for each document, and then trained a set of classifiers by adding the selected documents to TRN while holding the negativity assumption as true, i.e. we considered documents not marked as positive for a class as negative examples for training a classifier.

Table II
CLASSIFICATION RESULTS BY USING RELABELED DATA ONLY AS POSITIVE VS. BOTH POSITIVE AND NEGATIVE, FOR TOP-100 DOCUMENTS RELABELED FOR EACH CLASS USING IDHC CLUSTERING

Method	Micro-F_1	Macro-F_1
Documents from S as Positive-Only	0.7389	0.7471
Documents from S as Positive and Negative	0.6485	0.6512

We then tested this classifier on the held out test set and compared the results with a classifier that did not use the negativity assumption. Table II shows the result of this experiment. We observe a significant drop in both the Micro and Macro F_1 scores when the negativity assumption is used

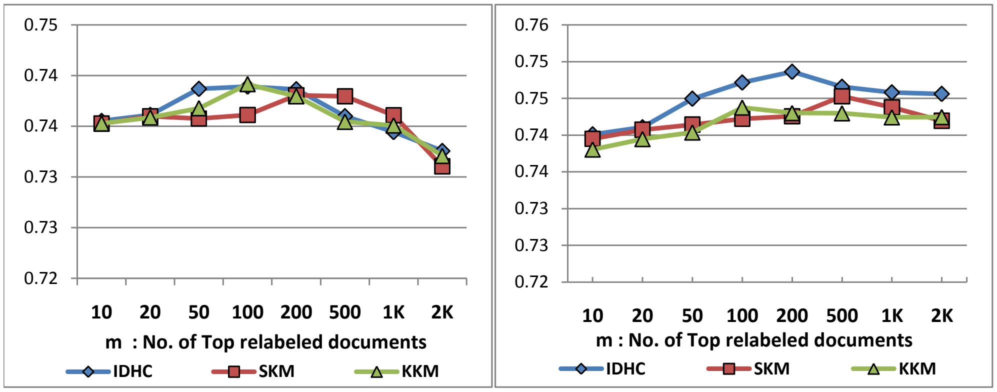

Figure 3. Comparing Micro-F_1 (left) and Macro-F_1 scores (right) on the held-out test data for various values of m using IDHC, Spherical k-means and Kernel k-means, when top-m selected documents for each class augmented the high-quality labeled training data. The baseline Micro-F_1 and Macro-F_1 scores (using the high-quality training data only) were 0.678 and 0.662 respectively

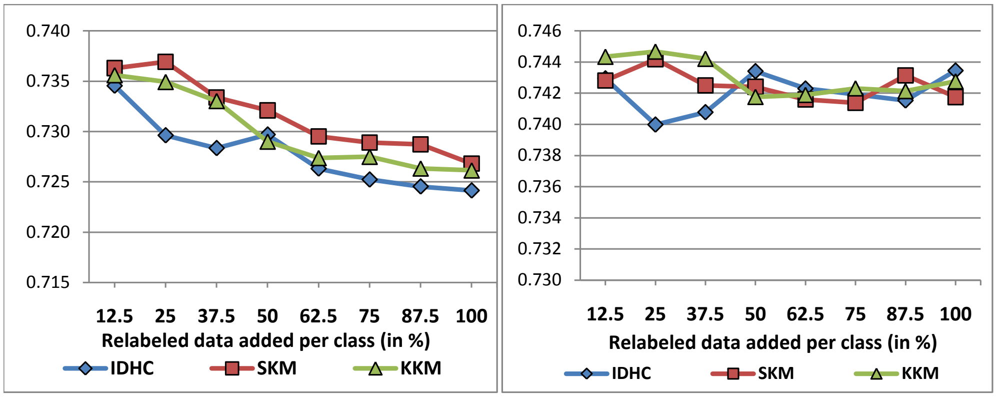

Figure 4. Comparing Micro-F_1 (left) and Macro-F_1 scores (right) on the held-out test data, by mixing high-quality labeled documents with additional documents from S, incrementally increasing the percentage of data added per class in each set

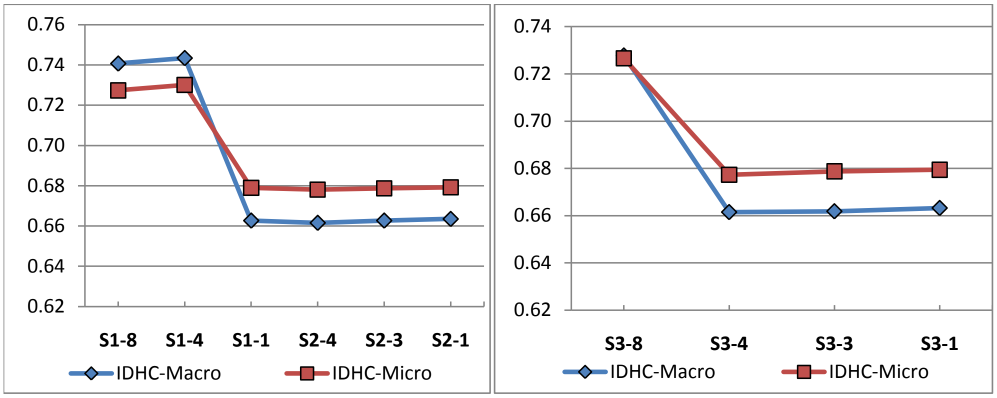

Figure 5. Comparing various evidence evaluation schemes. Each scheme is coded as S$a - b$, where Sa is described in section III-C and b value of the threshold t used for clustering. Macro and Micro $F1$ scores are reported on held out test data, using classifiers trained using documents relabeled using each scheme individually.

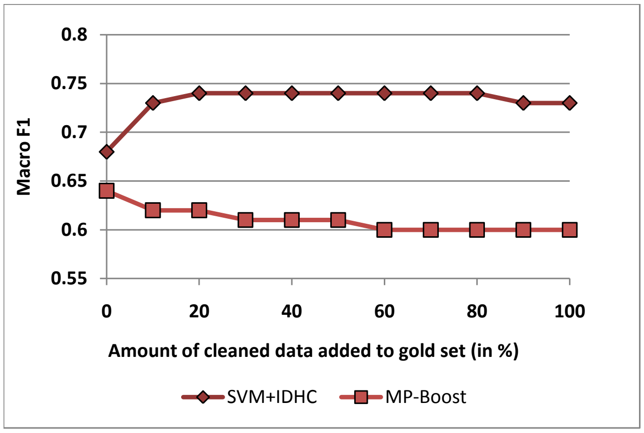

Figure 6. Change in Macro-F_1 as amount of cleaned data using MPBoost added to TRN is increased.

with the selected documents. Hence supporting our argument in Section III-B.

V. Conclusions and Future Work

In this paper we proposed ATDC, a novel training data cleaning method that uses training examples with high-quality class-labels to automatically validate and correct labels of noisy training data. Our method uses intuitive ways of automatically generating additional evidence from high-quality training data and uses simple evidence evaluation schemes to validate and select class-labels for noisy-labeled documents. We emphasize that using multiple sources of evidence for relabeling noisy documents provides better results as compared to a single source of evidence. We also argued that the common machine-learning assumption about the negativity of training documents with respect to classes that do not exist in their set of assigned class-labels should not be made for noisy-labeled datasets.

In the future, we plan to apply our method to other real-world datasets and experiment with additional clustering algorithms. We also plan to investigate additional evidence generation and evaluation schemes and to apply our method on non-textual data.

References

[1] S. Abney, R. E. Schapire, and Y. Singer. Boosting applied to tagging and pp attachment. In *Proceedings of the Joint SIGDAT Conference on Empirical Methods in Natural Language Processing and Very Large Corpora*, pages 38–45, 1999.

[2] V. Bhardwaj, R. J. Passonneau, A. Salleb-Aouissi, and N. Ide. Anveshan: A framework for analysis of multiple annotators' labeling behavior. 2010.

[3] G. Cong, W. S. Lee, H. Wu, and B. Liu. Semi-supervised text classification using partitioned em. In *11 th Int. Conference on Database Systems for Advanced Applications (DASFAA*, page 482493, 2004.

[4] A. Esuli and F. Sebastiani. Training data cleaning for text classification. In *Proceedings of the 2nd International Conference on Theory of Information Retrieval: Advances in Information Retrieval Theory*, ICTIR '09, pages 29–41, Berlin, Heidelberg, 2009. Springer-Verlag.

[5] R.-E. Fan, K.-W. Chang, C.-J. Hsieh, X.-R. Wang, and C.-J. Lin. LIBLINEAR: a library for large linear classification. *Journal of Machine Learning Research*, 9:1871–1874, 2008.

[6] F. Fukumoto and Y. Suzuki. Correcting category errors in text classification. In *Proceedings of the 20th international conference on Computational Linguistics*, COLING '04, Stroudsburg, PA, USA, 2004. Association for Computational Linguistics.

[7] D. Greene. *A State-of-theArt Toolkit for Document Clustering*. PhD thesis, 2006.

[8] E. Leopold and J. Kindermann. Text categorization with support vector machines. how to represent texts in input space? *Mach. Learn.*, 46:423–444, March 2002.

[9] H. H. Malik, J. R. Kender, D. Fradkin, and F. Moerchen. Hierarchical document clustering using local patterns. *Data Min. Knowl. Discov.*, 21:153–185, July 2010.

[10] A. McCallum and K. Nigam. A comparison of event models for naive bayes text classification. In *IN AAAI-98 WORKSHOP ON LEARNING FOR TEXT CATEGORIZATION*, pages 41–48. AAAI Press, 1998.

[11] A. K. Mccallum. Multi-label, multiclass document classification with a mixture model trained by em, 1999.

[12] K. Nigam, A. K. Mccallum, S. Thrun, and T. Mitchell. Text classification from labeled and unlabeled documents using em. In *Machine Learning*, pages 103–134, 1999.

[13] F. Sebastiani and C. N. D. Ricerche. Machine learning in automated text categorization. *ACM Computing Surveys*, 34:1–47, 2002.

[14] V. S. Sheng, F. Provost, and P. G. Ipeirotis. Get another label? improving data quality and data mining using multiple, noisy labelers.

[15] D. Vilar, M. J. Castro, E. Sanchis, and L. F. I. Vi. Multi-label text classification using multinomial models.

[16] J. T. yau Kwok. Automated text categorization using support vector machine. In *In Proceedings of the International Conference on Neural Information Processing (ICONIP*, pages 347–351, 1998.